(19) **RU** (11) **2 170 454** (13) **C2**

(51) МПК[7] **G 06 F 13/18, 15/00, 15/16**

РОССИЙСКОЕ АГЕНТСТВО
ПО ПАТЕНТАМ И ТОВАРНЫМ ЗНАКАМ

# (12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ РОССИЙСКОЙ ФЕДЕРАЦИИ

(21), (22) Заявка: **96120166/09, 27.12.1994**

(24) Дата начала действия патента: **27.12.1994**

(30) Приоритет: **07.03.1994 US 08/206,706**

(43) Дата публикации заявки: **27.12.1998**

(46) Дата публикации: **10.07.2001**

(56) Ссылки: **Y.Kistler and M.Satyanarayanan. Disconnected Operation in the Coda File System. ACM Transactions on Computer Systems. Vol 10, № 1, February 1992, p.3-25. SU 1304030 A1, 15.04.1987. M.Satyanarayanan, Y.Kistler et al. Coda: A Highlu Available File System for a Distributed Workstation Envirоument. IEEE Transaction on Computers. Vol. 39, № 4, April 1990, p.447 - 459. EP 301921 A2, 01.02.1989. US 5289540 A, 22.02.1994.**

(85) Дата перевода заявки PCT на национальную фазу: **25.09.1996**

(86) Заявка PCT:
**EP 94/04316 (27.12.1994)**

(87) Публикация PCT:
**WO 95/24685 (14.09.1995)**

(98) Адрес для переписки:
**129010, Москва, ул. Большая Спасская 25, стр.3, ООО "Городисский и Партнеры", Кузнецову Ю.Д.**

(71) Заявитель:
**ИНТЕРНЭШНЛ БИЗНЕС МАШИНЗ КОРПОРЕЙШН (US)**

(72) Изобретатель: **КЭНТРЕЛЛ Томас Джордж (US), ДЖАДЖИ Себнем (US), ШАХИН Амаль Ахмед (US), УОРД Ричард Байрон (US)**

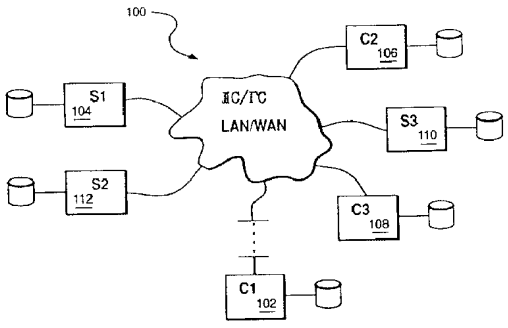
(73) Патентообладатель:
**ИНТЕРНЭШНЛ БИЗНЕС МАШИНЗ КОРПОРЕЙШН (US)**

(74) Патентный поверенный:
**Кузнецов Юрий Дмитриевич**

## (54) СИСТЕМА И СПОСОБ ЭФФЕКТИВНОГО ИСПОЛЬЗОВАНИЯ КЭШ-ПАМЯТИ В РАСПРЕДЕЛЕННОЙ ФАЙЛОВОЙ СИСТЕМЕ

(57)

Изобретение относится к системам обработки информации для управления данными. Техническим результатом является обеспечение возможности работы пользователя при отключении системы от спецпроцессора, эффективность обработки данных. Система содержит процессор, память, администратор кэш-памяти, кэш-память. Способ описывает работу данной системы. 2 с. и 4 з.п. ф-лы, 7 ил.




Фиг.1



RU 2170454 C2

RU 2170454 C2

(19) **RU** (11) **2 170 454** (13) **C2**

(51) Int. Cl.[7] **G 06 F 13/18, 15/00, 15/16**

RUSSIAN AGENCY
FOR PATENTS AND TRADEMARKS

## (12) ABSTRACT OF INVENTION

(21), (22) Application: **96120166/09, 27.12.1994**

(24) Effective date for property rights: **27.12.1994**

(30) Priority: **07.03.1994 US 08/206,706**

(43) Application published: **27.12.1998**

(46) Date of publication: **10.07.2001**

(85) Commencement of national phase: **25.09.1996**

(86) PCT application:
**EP 94/04316 (27.12.1994)**

(87) PCT publication:
**WO 95/24685 (14.09.1995)**

(98) Mail address:
**129010, Moskva, ul. Bol'shaja Spasskaja 25, str.3, OOO "Gorodisskij i Partnery", Kuznetsovu Ju.D.**

(71) Applicant:
**INTERNEhShNL BIZNES MAShINZ KORPOREJShN (US)**

(72) Inventor: **KEhNTRELL Tomas Dzhordzh (US), DZhADZhI Sebnem (US), ShAKhIN Amal' Akhmed (US), UORD Richard Bajron (US)**

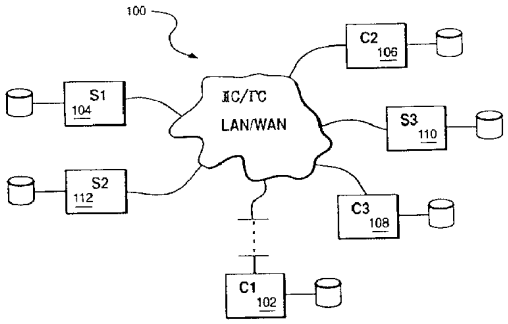
(73) Proprietor:
**INTERNEhShNL BIZNES MAShINZ KORPOREJShN (US)**

(74) Representative:
**Kuznetsov Jurij Dmitrievich**

# (54) SYSTEM AND METHOD FOR EFFECTIVE USE OF CACHE MEMORY IN DISTRIBUTED FILE SYSTEM

(57) Abstract:

FIELD: information processing systems for data control. SUBSTANCE: system has processor, memory, cache memory administrator, and cache memory. System user is enabled to operate even when system is disconnected from special processor unit. EFFECT: enhanced data processing efficiency. 6 cl, 7 dwg




Фиг.1

RU 2170454 C2

RU 2170454 C2

Данное изобретение относится к системам обработки информации для управления данными. Более конкретно: данное изобретение относится к управлению данными у пользователей в распределенной файловой системе. Еще конкретнее: данное изобретение относится к использованию кэш-памяти и ведению регистрации у пользователей распределенной файловой системы, которая может обеспечивать работу пользователя при подключении или отключении.

Известный уровень техники

Компьютерные автоматизированные рабочие места в последнее время повысили свою производительность и емкость. Первоначально автоматизированные рабочие места использовались одним оператором для выполнения одного или нескольких изолированных заданий. Повысившееся развертывание автоматизированных рабочих мест для многих пользователей в одной организации обусловило необходимость связи между автоматизированными рабочими местами и совместного использования пользователями. Это привело к разработке архитектур распределенных файловых систем, одна из которых изображена на фиг. 1.

Некоторое число автоматизированных рабочих мест на фиг. 1 подключено друг к другу через локальную сеть (ЛС). Глобальная сеть (ГС) может использоваться для взаимного подключения некоторого числа ЛС. ЛС и ГС составляют единую логическую сеть. Автоматизированные рабочие места на фиг. 1 обозначаются как спецпроцессоры S1, S2, S3, а пользователи - C1, C2, C3.

Обозначение автоматизированного рабочего места как пользователя или спецпроцессора (служебного файлового процессора) зависит от функции, выполняемой определенным автоматизированным рабочим местом в сети. То или иное автоматизированное рабочее место может быть и пользователем, и спецпроцессором. Выполнение распределенной файловой системы требует присутствия, по меньшей мере, одного спецпроцессора и, по меньшей мере, одного пользователя.

Автоматизированные рабочие места пользователей обычно подключаются к ЛС всегда для получения услуг ЛС и для совместного использования ресурсов, таких как файлы. Отключение от ЛС означало лишение автоматизированного рабочего места ресурсов ЛС. Работу можно было продолжать у отсоединенного пользователя только в том случае, если нужные файлы и программы были скопированы для пользователя до отключения.

Повысившаяся производительность портативных автоматизированных рабочих мест сделала практичной отсоединенную или "мобильную" работу с компьютером. Портативное автоматизированное рабочее место можно купить с такой производительностью процессора, памяти и запоминающего устройства на дисках, которая равнозначна используемым в настольных или офисных аппаратах. Портативные автоматизированные рабочие места или портативные компьютеры, тем не менее, в настоящее время имеют те же ограничения, от которых они страдают при отсоединении: разделяемыми ресурсами нельзя пользоваться, если копия файла или программы не сделана до отключения. Когда автоматизированное рабочее место подключается вновь, пользователю приходится самому "вручную" согласовывать файлы, измененные в портативном устройстве, с файлами сети. Фиг. 1 изображает отключаемое портативное автоматизированное рабочее место в виде пользователя C1 102, соединенного пунктирной линией с сетью.

Совместное использование файлов данных по сети, как показано на фиг. 1, выработалось с течением времени. Самая простая форма совместного использования дает возможность пользователю запрашивать данные у файла в спецпроцессоре. Необходимые данные направляются к процессору пользователя, а какие-либо изменения или модификации данных возвращаются к спецпроцессору. Создаются соответствующие блокировки для того, чтобы второй пользователь не изменил данные в файле, который находится у первого пользователя.

Распределенные файловые системы увеличивают масштаб совместного использования файлов при помощи дополнительных механизмов для более эффективного распределения данных между пользователями и для более эффективного управления совместным использованием файлов. Существует много распределенных файловых систем. Одной из популярных распределенных файловых систем является файловая Система "Эндрю" (АФС), которую распространяет корпорация "Трансарк".

АФС повышает производительность распределенной обработки данных путем создания файловой кэш-памяти у пользователя, который совершает доступ к данным спецпроцессора. Обращения к этой кэш-памяти делаются прикладными программами пользователя и только неудачное обращение в кэш-память может быть причиной того, чтобы данные приходилось бы доставать у спецпроцессора. Кэширование данных уменьшает поток обмена в сети и ускоряет время отклика для пользователя. Согласованность кэш-памяти АФС основана на системе обратного вызова спецпроцессора. Спецпроцессор уведомляет каждого пользователя о том, что у него есть определенные кэшированные данные того или иного состояния, которое делает недействительными кэшированные данные пользователя. Став недействительными, эти данные уничтожаются в кэш-памяти и должны приобретаться вновь у спецпроцессора при необходимости.

Отключение автоматизированного рабочего места, подключенного к АФС, вызовет потерю пользователем доступа к распределенным файлам. Он при отключении автоматически потеряет все обратные вызовы, из-за чего кэш-память станет неработоспособной.

Распределенная Файловая Система (РФС) является продолжением АФС, которая обеспечивает Распределенную Вычислительную Среду (РВС) ОСФ (Open Softaware Foundation). Распределенная файловая система использует основанный на работе спецпроцессора механизм опознавательного знака ("маркера") для



RU 2170454 C2

RU 2170454 C2

обеспечения согласованности кэш-памяти пользователя. Пользователи приобретают маркеры "считывания" для обеспечения действительности данных в своей кэш-памяти. Если пользователю приходится изменить данные в файле, он для данных приобретает маркер "записи". Выдача пользователю, например пользователю C1, маркера "записи" делает недействительными маркеры "считывания" для тех же данных у всех других пользователей.

Отмена маркера делает недействительными кэшированные данные у этих пользователей. При отключении от распределенной файловой системы пользователь лишается всех выданных маркеров. Из-за отсутствия маркеров он не сможет производить операции считывания или записи даже на кэшированных данных.

Университет Карнеги-Меллона сделал разработку для сохранения работоспособности отсоединяемых автоматизированных рабочих мест, подключаемых к сети АФС. Проект CODA направлен на обеспечение распределенной файловой системы постоянным наличием данных. Это делается путем точного копирования данных в спецпроцессорах и путем обеспечения сохранения работоспособности для отключенных операций пользователей. См.: "CODA: Файловая Система с Высокой Степенью Наличия для Работы Автоматизированных Рабочих Мест в Распределенной Среде", Труды Института инженеров по электротехнике и радиоэлектронике (IEEE), том 39, N 4, апрель 1990.

Распределенная работа в Coda выполняется оптимистическим управлением использования копий данных. Оптимистическое управление предоставляет многим пользователям возможность считывать и записывать данные даже когда они отключены. Несогласованности данных оставляются для последующего идентифицирования и разрешения. Пессимистическое управление использованием копий данных избегает все конфликты путем ограничения считывания и записи одним разделом. Отключение лишает пользователя возможности определять, имеет ли доступ к данным другой пользователь. При этом пессимистическом управлении использованием копий данных пользователь не сможет осуществлять запись. См.: "Работа при Отключении в Файловой Системе Coda", Дж.Кистлер, М. Сатьянарайанан, Университет Карнеги-Меллона, Протоколы 13 Симпозиума АВТ по Принципам Операционных Систем, октябрь 1991.

Распределенная работа в системе Coda позволяет пользователю продолжать осуществлять доступ к данным и обновлять данные в кэш-памяти пользователя. Файловые операции не срабатывают только в том случае, когда происходит неудачное обращение в кэш-память, т.к. только в этом случае пользователь не в состоянии совершать доступ к данным из спецпроцессора. АФС и Coda кэшируют файлы полностью. Coda обновляет файл во время отключенных операций и ведет реестр всех изменений данных. При повторном подключении пользователь обязан обновить все спецпроцессорные копии данных при помощи применения зарегистрированных транзакций для файлов спецпроцессора. Несогласованности данных обрабатываются отложенным решением для вмешательства "вручную".

Основным недостатком Coda является то, что сохраняется только подключение к спецпроцессору Coda. Аналогично АФС, Coda является протоколом распределенной файловой системы.

Coda основана на том, что спецпроцессор или спецпроцессоры и все пользователи используют один и тот же протокол распределенной файловой системы. Для этого требуется, чтобы все пользователи и спецпроцессоры в сети перешли к протоколу Coda для поддержания отключенной работы.

Второе техническое решение предложено Хастоном и Ханимэном в "Работа при Отключении для АФС", Л.Б.Хастон и П.Ханимэн, Центр Интеграции Информатики, Мичиганский Университет, опубликовано в Протоколах Симпозиума USENIX Мобильной и Автономной по Местоположению Вычислительной Техники, август 1993. Хастон и Ханимэн предлагают систему пользователей, которая может подключаться к стандартному спецпроцессору АФС, не требуя при этом модификаций для спецпроцессора. Модификации в коде пользователя поддерживают отключенную работу и повторное подключение. Повторное подключение и повторное согласование выполняются регистрацией каждой транзакции при отключении. Каждое считывание, запись или обновление записываются в реестре транзакций, который при повторном подключении воспроизводится для спецпроцессора. Это техническое решение имеет преимущество поддержания работы стандартного спецпроцессора АФС. Но это зависит от наличия протокола распределенной файловой системы АФС, и этот метод не может действовать с другими спецпроцессорами, отличающимися от спецпроцессора АФС. Метод всеобщей регистрация также наносит ущерб производительности системы. Объем регистрирования также создает трудности с емкостью диска пользователя и обусловливает задержки согласования в системах любого размера.

В журнале IEEE "Техника Программного Обеспечения", том 19, N 6, июнь 1993, стр. 613-624, "Доступ к Файлам в "Интернете": Файловая Система "джейд", Х.Рао и др. описывают кэшированную файловую систему для использования в условиях ЛС или "Интернет". Рао и др. выполняют все файловое кэширование файлов в автоматизированном рабочем месте или на месте, выбранном пользователем. Файловая система "Джейд" требует, чтобы разработчик прикладной программы использовал "Единый Интерфейс для Протокола Доступа". Для этого нужно модифицировать прикладные программы или, как предлагается в статье, ввести совместно используемую библиотеку, чтобы заменить стандартные вызовы Едиными интерфейсными вызовами. Система "Джейд" использует "посредников" файловой системы, чтобы делать фактические запросы файловой системы. "Джейд" не предоставляет возможности поддержания отключенной работы и регистрации действий файловой



RU 2170454 C2

RU 2170454 C2

системы для синхронизации после повторного включения. "Джейд" не обеспечивает средство перехвата стандартных вызовов файловой системы и преобразования их в независимый от разделителя синтаксис перед доступом к администратору кэш-памяти.

"Компьютерные Стандарты и Интерфейсы", том 14, N 3, 1992, стр. 191-208, А. Маршалл и др. "Осуществимость Файловых спецпроцессоров на базе ISO-FTAM для Выполнения Однородной Файловой системы". Маршалл и др. описывают техническое решение однородных файловых систем на основе модели ISO-FTAM. Они критикуют сегодняшние технические решения: "Sun NFS", например, в связи с тем, что они основаны на решениях, зависимых от операционных систем. Данное изобретение обеспечивает независимое от операционной системы средство для использования виртуальных узлов сети (VNodes) в качестве стандартного протокола, очень близкого к FTAM. Данное осуществление устраняет зависимость от команд операционной системы. Виртуальные файловые запоминающие устройства (ВФЗУ) Маршалла и др. не поддерживают доступ к имеющимся распределенным файловым системам. Предлагаемая файловая система заменяет имеющиеся структуры, хотя доступ к стандартным физическим типам файла поддерживается. Технической проблемой остается обеспечение однородного механизма доступа, который не требует замены существующих протоколов распределенной файловой системы.

Протоколы 3-го Совещания по Операционным Системам Автоматизированных Рабочих Мест, 23 апреля 1992, стр. 122-125, М. Сатьянарайанан и др. "О Повсеместности Регистрирования в распределенных Файловых Системах". Эта статья описывает регистрационные функции в описанной выше системе Coda. Регистрирование в Coda обеспечивает воспроизведение транзакций во время отключения. Как указывалось выше, для CODA требуется спецпроцессор с протоколом CODA доступа к распределенной файловой системе. Регистрирование поэтому основывается на однородной модели спецпроцессора и поддержание однородной модели осуществляется им не без труда.

Поэтому имеется техническая проблема обеспечения распределенной файловой системы, которая обеспечит непрерывную работу "отключенных" пользователей с эффективным согласованием при повторном подключении. Еще одной технической проблемой является создание пользователя распределенной файловой системы, которого можно будет использовать при помощи многочисленных протоколов распределенной файловой системы. Наконец, есть и техническая проблема создания пользователя распределенной файловой системы, который будет независим от синтаксиса операционной системы для разрешения объекта или полного имени файла.

Сущность изобретения

Данное изобретение направлено на обеспечение распределенной файловой системы, которая обеспечивает как подключенную, так и отключенную работу пользователя. Файловая система пользователя выполнена с возможностью подключения к нескольким архитектурам и обеспечивает эффективную регистрацию транзакций.

Данное изобретение направлено на способ управления кэш-памятью файловой системы в пользовательской компьютерной системе, действующей под управлением первой операционной системы. Способ содержит следующие этапы: перехват запросов операционной системы для объекта файловой системы в распределенной файловой системе; преобразование указанных запросов для удаления синтаксиса, зависимого от операционной системы; проверка кэш-памяти в средстве запоминающего устройства указанного пользователя на присутствие данных об указанном объекте файловой системы; выполнение указанного запроса объекта файловой системы - если данные кэш-памяти существуют.

Поэтому задача данного изобретения заключается в обеспечении пользователя распределенной файловой системы, который может работать при отключении от спецпроцессора.

Другая задача данного изобретения заключается в обеспечении отключаемого пользователя, который эффективно регистрирует транзакции для согласования при повторном подключении.

Еще одна задача данного изобретения заключается в обеспечении пользователя файловой системы эффективным использованием кэш-памяти в целях уменьшения объема памяти и пространства диска, требуемых для кэширования, и для увеличения скорости и эффективности обработки данных в кэш-памяти.

Еще одна задача данного изобретения заключается в обеспечении пользователя распределенной файловой системы, который может работать с несколькими протоколами распределенной файловой системы.

Еще одна задача данного изобретения заключается в обеспечении пользователя распределенной файловой системы, который будет независим от операционной системы.

И еще одна задача данного изобретения заключается в повышении эффективности кэш-памяти путем "агрессивного" кэширования разрешенных названий объекта.

Еще одна задача данного изобретения заключается в обеспечении системы управления кэш-памятью, которая не зависит от названия объекта, чтобы каждый объект файловой системы мог бы иметь нескольких родителей, каждый из которых имеет различные правила синтаксиса названия объекта.

Вышеуказанные и другие задачи, признаки и преимущества данного изобретения будут очевидны из следующего ниже более подробного описания предпочтительного варианта реализации данного изобретения, иллюстрируемого прилагаемыми чертежами, в которых аналогичные цифровые обозначения представляют аналогичные компоненты объектов изобретения.

Краткое описание чертежей

Фиг. 1 - иллюстрация среды распределенной сети, в которой осуществляется данное изобретение.

Фиг. 2 - блок-схема автоматизированного рабочего места, изображающая один из



RU 2170454 C2

RU 2170454 C2

вариантов реализации данного изобретения.

Фиг. 3 - блок-схема известного уровня техники распределенной файловой системы.

Фиг. 4 - блок-схема, изображающая функциональные компоненты варианта реализации данного изобретения.

Фиг. 5 - изображение структур данных в предпочтительном варианте реализации данного изобретения.

Фиг. 6 - блок-схема варианта реализации данного изобретения.

Фиг. 7 - схема последовательности, изображающая этапы процесса одного из вариантов реализации данного изобретения.

Подробное описание

Фиг. 2 изображает типичную конфигурацию пользователя или автоматизированного рабочего места спецпроцессора. Данное изобретение предпочтительно осуществляется с таким автоматизированным рабочим местом, как IBM RISC System/6000 или персональный компьютер IBM PS/2. Автоматизированные рабочие места пользователя и спецпроцессора взаимосвязаны "маркерным" звонком или ЛС "Эсернет" - изображено на фиг. 1. Автоматизированные рабочие места, подключенные к ЛС, могут включать в себя портативные автоматизированные рабочие места, такие как С1, которые могут быть отключены от ЛС и использоваться независимо от ЛС. Будет очевидно, что в рамках данного изобретения можно использовать много других конфигураций аппаратуры автоматизированного рабочего места или типов ЛС.

Обычное автоматизированное рабочее место 200 пользователя или спецпроцессора имеет процессор 210, память системы 214 и энергонезависимое запоминающее устройство 212, такое как НМД, дискета или оптическое запоминающее устройство. Процессор принимает входной сигнал от устройств ввода-вывода, таких как клавиатура 224 и координационно-указательное устройство 226 через контроллер ввода-вывода 218. Система представляет на дисплее 222 оператору графическую информацию, управляемую контроллером 220 графики. Автоматизированное рабочее место подключено к сети (не показана) через адаптер 216 интерфейса сети.

Предпочтительный вариант реализации данного изобретения выполняется компьютерным процессом, действующим в памяти и процессоре автоматизированного рабочего места 200. Компьютерное программное изделие, осуществляющее данное изобретение, могут хранить в энергонезависимом запоминающем устройстве 212, включая хранение на ленте, дискете или устройстве КД-ПЗУ.

Работа распределенных файловых систем известного уровня техники иллюстрируется со ссылкой на фиг. 3. Фиг. 3 изображает автоматизированное рабочее место 302 пользователя и автоматизированное рабочее место 304 спецпроцессора (служебного файлового процессора), взаимно подключенные сетевым интерфейсом 306. Запрос пользовательского процесса 308 о конкретном файле или информации о файле, постоянно находящейся на дисковом запоминающем устройстве 390, обрабатывается следующим образом.

Обратите внимание на то обстоятельство, что и пользователь 302, и спецпроцессор 304 разделены на пространство 310, 314 адреса пользователя и пространство 312, 316 адреса ядра. Интерфейсом между прикладной программой и ядром (известен как интерфейс прикладной программы или ИПП) является ИПП-виртуальная файловая система/узел сети VFS/vnode (VFS/vnode), который определяется в: С.Р.Клаймэн, "Vnodes: Архитектура для Типов Множественных Файловых Систем в Sun UNIX", Протоколы Летней Конференции Usenix, 1986". Можно также, разумеется, использовать другие интерфейсы. Интерфейс VFS/vnode дает возможность существовать множественным виртуальным файловым системам. Распределенная файловая система становится виртуальной файловой системой для автоматизированного рабочего места пользователя.

Пользовательский процесс 308 запрашивает файл данных в запоминающем устройстве 390. Запрос 320 перехватывается интерфейсом VFS и поступает к пользователю 322 распределенной файловой системы, постоянно находящемуся в пространстве 312 адреса ядра пользователя. Пользователь распределенной файловой системы управляет связью со спецпроцессором по сети. Запрос подается в сообщении 326 спецпроцессору 324 распределенной файловой системы в ядро 316 спецпроцессора по сети 306. Спецпроцессор 324 распределенной файловой системы совершает доступ к запоминающему устройству 390 с помощью интерфейса 328 VFS для доступа к локальной физической файловой системе спецпроцессора, которая содержит устройство 390. Запрошенные данные приобретаются и направляются 330 от спецпроцессора 324 распределенной файловой системы к пользователю 322 распределенной файловой системы, который в свою очередь направляет 332 данные к запрашивающему пользовательскому процессу 308.

Пользователь 322 распределенной файловой системы может включить кэш-память данных (не показана) в данные кэш-памяти, запрошенные со спецпроцессора. Перед запросом данных от спецпроцессора пользователь распределенной файловой системы проверяет кэш-память относительно действительных данных.

Предпочтительный вариант реализации данного изобретения иллюстрируется на фиг. 4. Автоматизированное рабочее место пользователя имеет пространство 402 адреса пользователя и пространство 404 адреса ядра. Усовершенствованный администратор 406 кэш-памяти в данном изобретении действует предпочтительно в пространстве 402 адреса пользователя, но может быть выполнен в пространстве 404 адреса ядра.

Программы пользователя 408, 410, 412 дают запросы 414, 416, 418 файловой системы. Программа n пользователя изображена использующей логическую файловую систему 414, которая поочередно дает запросы 420 файловой системы. Запросы файловой системы соответствуют интерфейсу VFS/vnodes, хотя в рамках данного изобретения могут использовать и другие интерфейсы. Данное изобретение можно использовать в любой логической файловой



RU 2170454 C2

RU 2170454 C2

системе, которая соответствует ИПП-VFS/vnode или в качестве вариантов осуществления соответствует любому другому ИПП.

Интерфейс VFS направляет запрос файловой системы администратору 406 кэш-памяти. Администратор 406 кэш-памяти хранит локальную информацию о распределенных файловых системах, к которым пользователь совершает доступ. При возможности запросы файловой системы обслуживаются администратором кэш-памяти без доступа к спецпроцессору. Запросы файловой системы могут выполняться администратором кэш-памяти, если запрошенная информация находится в кэш-памяти и все еще действительна. Администратор 406 кэш-памяти будет описан более подробно ниже.

Запросы файловой системы выполняются администратором 406 кэш-памяти либо с данных, постоянно находящихся в памяти администратора кэш-памяти, или доступом к локальной физической файловой системе 420, либо доступом 422 к распределенной файловой системе. Администратор кэш-памяти хранит локальные копии распределенных файлов в локальной физической файловой системе в целях быстрого доступа. Доступ 422 к распределенной файловой системе отдает необходимые команды к спецпроцессору 324 распределенной файловой системы - как описывается выше. Предпочтительное осуществление кэширует все файлы, хотя данное изобретение не ограничивается полным кэшированием файлов.

Администратор 406 кэш-памяти и доступ 422 к распределенной файловой системе не зависят от операционной системы и протокола доступа к распределенной файловой системе. Независимость операционной системы достигается обеспечением указанных интерфейсов. Обеспечиваемые интерфейсы: интерфейс VFS+/Vnodes для операций файловой системы; интерфейс последовательности, такой как предоставляемый структурой последовательности объектов IBM SOB; локальный интерфейс файловой системы (ЛФИ) для кэшированных файлов; различные протоколы распределенной файловой системы; интерфейс синхронизации кэш-памяти; интерфейс воспроизведения регистрации. Администратор кэш-памяти не зависит от какого-либо синтаксиса операционной системы. Названия файлов преобразуют, чтобы они стали независимыми от знаков разделителя компонентов названия файла и считались не имеющими зарезервированных знаков и зарезервированных слов. Определение маршрутного имени файла также независимо от синтаксиса операционной системы. Поэтому к конкретному объекту файловой системы можно обращаться двумя операционными системами с противоречащими правилами синтаксиса. Например, администратор 406 кэш-памяти может быть выполнен для работы с операционной системой IBM OS/2, или IBM AIX, или с любой другой операционной системой лишь с незначительными модификациями.

Администратор кэш-памяти в данном изобретении не зависит от протокола распределенной файловой системы. Доступ 422 к распределенной файловой системе может совершить доступ к удаленным спецпроцессорам либо с помощью спецпроцессора Распределенной Вычислительной Среды, который поставляет "Оупен Софтвзар Фаундейшн" (ОСФ) или с помощью программного изделия IBM LAN Server. Другие протоколы доступа к удаленным файлам можно использовать только с незначительными модификациями. Механизм синхронизации и механизм воспроизведения реестра обрабатывают зависимые от протокола задания.

Администратор 406 кэш-памяти ведет кэш-память 409 пользователя. В связи с кэш-памятью 409 администратор кэш-памяти ведет информацию 410 о томе и реестр 412 модификаций. Информация 410 тома содержит информацию о файловых системах, доступных для автоматизированного рабочего места пользователя через администратора кэш-памяти. В предпочтительном осуществлении удаленные файловые системы подключают к пользователю посредством тома. Каждый том будет иметь вход тома в базе данных информации. Итак, удаленная файловая система будет подключена к автоматизированному рабочему месту пользователя либо как файловая система типа Unix (например, установленная в качестве конкретной файловой системы/rfs), либо как символ накопителя Распределенной Вычислительной Среды (например, как пуск G:\.). Вся информация об удаленной файловой системе и ее состоянии хранится в базе данных 410 информации тома. Множественные удаленные файловые системы можно одновременно подключать к одному пользователю.

Структура кэш-памяти 409 изображена более подробно на фиг. 5. Кэш-память 409 ведет единую кэш-память для полного названия файла, данных объекта файловой системы и информации состояния объекта файловой системы. Кэш-память ведется в последовательной динамически распределяемой области памяти, т.е. она периодически записывается в энергонезависимое запоминающее устройство, из которого она может быть восстановлена в случае случайного отключения электроэнергии или отказе системы. Единая интегрированная кэш-память сокращает общие требования к кэш-памяти, тем самым улучшая показатели работы кэш-памяти. Для единой кэш-памяти требуется только одна рандомизированная таблица вместо трех, которые применяются в системах имеющегося уровня техники. Всю обработку данных кэш-памяти можно выполнять в одном кодовом наборе, который хранит различные типы объектов файловой системы, а не в обычном механизме отдельного кода для каждого типа кэш-памяти. Кэшированный объект файловой системы в общем виде изображен под обозначением 502. Объект 502 файловой системы используется как для подключенной, так и неподключенной работы администратора кэш-памяти.

Объект 502 файловой системы является структурой данных, содержащей информацию, необходимую для доступа к удаленному файлу или локальной кэшированной копии





RU 2170454 C2

удаленного файла. Элемент объекта файловой системы создается в кэш-памяти для каждого удаленного файла или каталога, к которым обращается автоматизированное рабочее место пользователя. Объект файловой системы содержит идентификатор 504 VFS+/vnode и идентификатор 506 файла.

Идентификатор 506 файла включает в себя данные тома, спецпроцессора и файла. Идентификатор VFS+/vnode содержит информацию, определяемую стандартной спецификацией VFS/vnodes. Включение этой информации позволяет полностью поддерживать все интерфейсы VFS/vnode. В данном изобретении дополнительная информация прилагается к структуре данных для обеспечения увеличенного кэширования и работы при отключении. Элемент 508 тома является указателем элемента 545 тома в базе данных 410 информации тома для тома, к которому относится данный файл. Переменная 510 состояния и структура 512 состояния ведутся для предоставления для данных информации состояния кэш-памяти. В предпочтительном осуществлении, выполненном в ориентированной на объект среде, с помощью таких средств, как язык программирования C++, структура состояния определена как класс объекта. Структура состояния для конкретного файла операционной системы (например, OS/2 или AIX) определяется созданием подкласса класса структуры состояния. Точным содержанием, необходимым для определения состояния файла, подкласс переопределяет класс структуры состояния.

Объект файловой системы ведет информацию 514 когерентности кэш-памяти. Эта информация используется для обеспечения согласованности между кэш-памятью и спецпроцессором и точными копиями спецпроцессора. Список перечней 516 управления доступом определяет разрешения на доступ к файловой системе и на модификации согласно нормативу IEEE POSIX. Другая флаговая информация ведется в 518. Состояние установления тома ведется 520 в качестве указателя 522 на родительский элемент и информации 524 приоритета. Каждый объект файловой системы имеет присвоенный приоритет, который используется при определении объектов на уничтожение, чтобы дать место новым объектам.

Доступ к данным файловой системы производится через указатель 526 файла/каталога/символьного звена. Этот указатель обеспечивает доступ к трем разным типам данных кэш-памяти в зависимости от типа объекта файловой системы. Объект файловой системы для файла данных имеет указатель на кэшированный файл 540, который в свою очередь указывает на копию локального файла кэшированных данных в локальной файловой системе 420. Указатель каталога показывает на кэшированный каталог 542. Кэшированный каталог 542 выполнен на основе схемы агрессивного кэширования. Когда для конкретного каталога запрашивается информация о каталоге, то этот каталог и все его непосредственные дочерние записи возвращаются. Это уменьшает количество неудачных обращений в кэш по причине большой вероятности того, что информация об одной или нескольких дочерних записях будет запрошена вскоре после информации о родительском каталоге. Кэш-память каталога предоставляет информацию, необходимую для определения маршрутного имени файла, т.е. отображения конкретного запроса маршрута файловой системы на определенном файле данных. Наконец, указатель 526 может указывать на кэшированную информацию 544 символьного звена. Эта кэшированная информация обеспечивает разрешение символьных звеньев.

В 528 обеспечиваются элементы для разных счетчиков, блокировок и "отмычек" (ручек) и индикатора чистого состояния 532. Индикатор 532 чистого состояния важен для работы при отключении и используется для указания состояния согласования.

Важно отметить, что название объекта файловой системы не включается в структуру 502 данных. Это позволяет бесконфликтно обращаться к объекту файловой системы от различных наборов названий объекта. У каждого объекта файловой системы может быть несколько родителей. Каждый родитель может соответствовать разным порядкам присваивания имен, например: спецификации присваивания имен Группы Управления Объектами (ГУО), или спецификации AIX, OS/2, либо в соответствии с другой операционной системой. Поэтому доступ к конкретному "списку сотрудников" (employee-list) объекта файла можно осуществлять с OS/2 как c: \ employee-list и с AIX как /usr/ employee-list.

Администратор 406 кэш-памяти обеспечивает как "подключенную", так и "отключенную" работу. Во время подключенной работы администратор 406 кэш-памяти выполняет запросы файловой системы от кэш-памяти или, в случае необходимости, путем доступа к удаленному файлу. При отключенной работе администратор кэш-памяти выполняет запросы от кэш-памяти и ведет реестр 546 модификаций по всем модификациям в файловой системе.

Один из вариантов выполнения администратора кэш-памяти в соответствии с данным изобретением изображен на фиг. 6. Это осуществление разработано для использования с операционной системой IBM 0S/2. Запросы файловой системы отдаются системой как команда 602 "doscall". Все запросы файловой системы обслуживаются файловой системой IFS. IFS обеспечивает логические возможности файловой системы. IFS может устанавливать и обслуживать различные типы файловых систем. Установленные файловые системы могут включать в себя, например, файловую систему 606 FAT, которая является общей для дисковых операционных систем, или файловую систему OS/2 HPFS. В условиях ЛС Инициатор Запроса (Requester) 610 ЛС фигурирует в качестве еще одной файловой системы, в которой доступы к файлам направляются к Спецпроцессору ЛС для исполнения. Администратор кэш-памяти согласно данному изобретению может устанавливаться как еще один тип файловой системы 612. Все доступы к удаленной файловой системе поступают к администратору 612 кэш-памяти. Администратор кэш-памяти проверяет



RU 2170454 C2

RU 2170454 C2

кэш-память 614 (в пространстве адреса пользователя), чтобы определить: кэширована ли запрашиваемая информация. Если да, то информация предоставляется прикладной программе, включая, при необходимости, предоставление или модифицирование кэшированных данных, удерживаемых в локальной файловой системе 606 или 608. Если данные файловой системы не обнаруживаются, то запрос 615 направляется к инициатору запроса 610 ЛС, чтобы он совершил доступ к нужным данным в спецпроцессоре ЛС. Когда удаленные данные поступят, они размещаются в кэш-памяти для будущих обращений.

Схема, иллюстрирующая последовательность процесса для команды записи файла, изображена на фиг.7. Другие команды файловой системы выполняются аналогично. Команда начинается в 702 - отдается команда 704 записи файла. Администратор кэш-памяти проверяет: есть ли в кэш-памяти 706 запрошенный файл. Если есть, то кэшовая копия файла обновляется 708. Затем кэш-память проверяют, чтобы определить: подключен 710 ли в данное время том. Состояние подключения поддерживается в базе данных 410 информации тома. Если том подключен, то изменение подается обратно к спецпроцессору 712 и обработка данных завершается 722. Если том не подключен, то изменение регистрируется 714 в реестре модификаций и обработка данных завершается 722.

Если запрошенного файла в кэш-памяти нет, то администратор кэш-памяти затем проверяет: подключен 716 или не подключен запрошенный том. Если подключен, то файл запрашивается у удаленной файловой системы 718, размещенной в кэш-памяти, и обработка данных возобновляется на этапе 708. Если том не подключен, то сигнал сбоя возвращается в прикладную программу 720.

Регистрирование видоизменений (например, этап 714) в реестре 546 осуществляется объектами регистрирования. Область проблемы представлена классом реестра модификаций пользователя и отдельным классом для каждого отличного от других типа регистрируемой транзакции. Для каждого вида работы определяют разные объекты, например, один будет для файла записи, создания файла, удаления файла. Когда администратор кэш-памяти регистрирует изменение 714 в отключенном режиме работы, создается тип объекта регистрирования, соответствующий запрашиваемой операции (например, запись файла). Для помещения объекта в реестр активизируют метод регистрации.

Объектные методы включают в себя методы уплотнения и оптимизации реестра для воспроизведения. Регистрирование конкретного объекта может обусловить удаление ранее зарегистрированного объекта. Например, когда файл создается, а затем удаляется, то объект удаления файла удалит элемент регистрации создания файла, все элементы регистрации модификаций файла и затем выйдет, не размещая в реестре элемент удаления файла. Эта система обеспечивает быструю и эффективную синхронизацию кэш-памяти с удаленной файловой системой при повторном подключении. Логические функции в объектах регистрирования устраняют необходимость обработки и уплотнения реестра перед воспроизведением.

Из изложенного выше описания будет ясно, что различные модификации и изменения можно производить в предпочтительном варианте реализации данного изобретения, не выходя за его рамки. Данное описание изложено в целях иллюстрации и не должно истолковываться в ограничивающем смысле. Объем патентной защиты данного изобретения определяется только его формулой изобретения.

## Формула изобретения:

1. Способ управления кэш-памятью файловой системы в пользовательской компьютерной системе, работающей под управлением операционной системы, чтобы выполнить запрос операционной системы, включающий в себя одну из множества операций файловой системы в качестве запрашиваемой операции файловой системы и имя файла распределительной файловой системы, отличающийся тем, что перехватывают запрос операционной системы, удаляют из имени файла синтаксис разделителя, зависимый от операционной системы, для создания обращения к объекту файловой системы, проверяют кэш-память файловой системы в средстве памяти пользовательской компьютерной системы на наличие объекта файловой системы для обращения к объекту файловой системы, причем объект файловой системы содержит информацию, позволяющую доступ к файлу распределенной файловой системы и доступ к возможным локальным кэшированным данным файла внутри кэш-памяти файловой системы, и выбирают один из множества возможных протоколов распределенной файловой системы и, если не имеется локальных кэшированных данных файла и есть подключение к распределенной файловой системе, запрашивают данные файла и запоминают их в качестве локальных кэшированных данных, выполняют запрошенную операцию файловой системы, если имеются локальные кэшированные данные файла.

2. Способ по п.1, отличающийся тем, что всякий раз, когда обнаруживают отсутствие подключения к распределенной файловой системе, дополнительно осуществляют следующие этапы: определяют тип запрошенной операции файловой системы, выполняемой на файле, создают запрашиваемый объект файла регистрирования согласно типу запрашиваемой операции файловой системы, активизируют метод обработки реестра модификаций для регистрирования запрашиваемого объекта файла регистрирования и регистрируют запрошенный объект файла регистрирования путем выполнения оптимизации реестра для определенного типа запрашиваемой операции файловой системы.

3. Способ по одному из п.1 или 2, отличающийся тем, что дополнительно запоминают созданное обращение к объекту файловой системы для каждого файла или каталога, к которым обращаются в упомянутой кэш-памяти файловой системы.

4. Пользовательская компьютерная система, подключенная через сеть, по меньшей мере, к одной обслуживающей



RU 2170454 C2

RU 2170454 C2

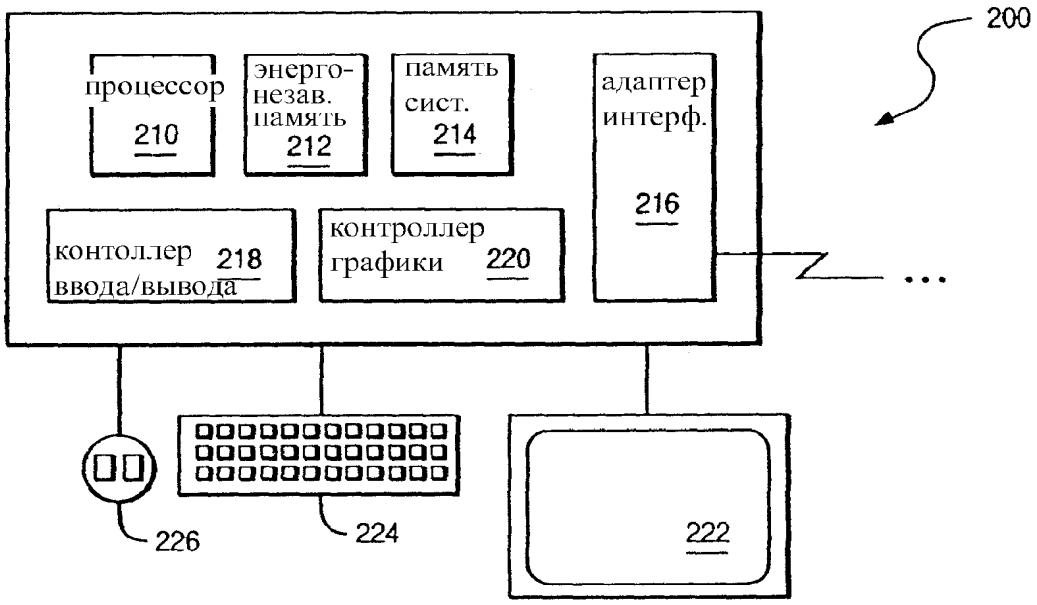
компьютерной системе, причем пользовательская и обслуживающая компьютерные системы содержат процессор и память и используют распределенную файловую систему, пользовательская компьютерная система выполнена с возможностью работы под управлением операционной системы, чтобы выполнить запрос операционной системы, включающий в себя одну из множества операций файловой системы в качестве запрашиваемой операции файловой системы и имя файла распределенной файловой системы, отличающаяся тем, что включает в себя администратор кэш-памяти, предназначенный для выполнения запроса файловой системы, выполненный с возможностью удаления из имени файла синтаксиса разделителя, зависимого от операционной системы, для создания обращения к объекту файловой системы, и кэш-память файловой системы в памяти пользовательской компьютерной системы для проверки наличия объекта файловой системы для обращения к объекту файловой системы, причем объект файловой системы включает в себя информацию, позволяющую доступ к файлу распределенной файловой системы и доступ к возможным локальным кэшированным данным файла внутри кэш-памяти файловой системы, причем кэш-память файловой системы выполнена с возможностью выбора одного из множества возможных протокольных средств распределенной файловой системы и, если не имеется локальных кэшированных данных файла и есть подключение к распределенной файловой системе, запроса данных файла и запоминания этих данных в качестве локальных кэшированных данных, причем кэш-память файловой системы выполнена с возможностью выполнения запрашиваемой операции файловой системы, если имеются локальные кэшированные данные файла.

5. Пользовательская компьютерная система по п.4, отличающаяся тем, что дополнительно включает в себя реестр модификаций объектов, управляемый администратором кэш-памяти таким образом, что, когда не обнаруживается подключение к распределенной файловой системе, администратор кэш-памяти регистрирует объект регистрирования, соответствующий запрашиваемой операции в реестре модификаций объектов.

6. Пользовательская компьютерная система по одному из п.4 или 5, отличающаяся тем, что дополнительно включает указатель файла (каталога) символьного звена, обеспечивающий доступ к трем разным типам данных кэш-памяти в зависимости от типа объекта файловой системы: к копии локального файла кэшированных данных в локальной файловой системе в памяти, кэшированному каталогу, к символьному звену кэш-памяти.

RU 2170454 C2

RU 2170454 C2




Фиг.2



RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2

RU 2170454 C2